INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

Course Name: Computer Systems Management

Course Code: CSE 131

Project: Chatting App Using C with TCP Implementation

App Name: iChat

Built by: Ankur Sharma

Roll No.: 2016225

Section: B

Group: 2

S. No.	Content
1.	Introduction with objective of the project
2.	Prerequisites for developing “iChat- A Chatting App”
3.	Technology used
4.	Methodology
5.	Snapshots of iChat
6.	Contribution

Introduction with objective of the project:

The aim is to develop a free chatting app which uses client server model that can be used in any IP domain or network, for example: IIITD Network Domain. Any two persons can send messages and receive messages from/on Terminal on different machine from both the sides for free using "iChat" developed by me. One more thing is that it is very light weight and Operating System Independent.

Prerequisites for developing "iChat- A Chatting App":

1> Socket Programming: Understanding of stream IO, managing data packets, creating sockets, accepting connection, listen to connections.

2> Networking: Basic information about TCP, UDP.

3> C Language: Knowledge of C libraries for socket programming.

Technology used:

Technology used is socket programming. Sockets provide the communication between two machines using TCP. A client program creates a socket on its end of the communication and tries to connect that socket to a server. So, basically it is inter-process communication which uses client-server model. When the connection is successfully established, the server creates a socket object on its end of the communication. The client and the server can now communicate by writing to and reading from the socket. TCP is a two-way communication protocol, hence data can be sent across both streams at the same time. Following are the useful classes providing complete set of methods to implement sockets.

Methodology:

- Terminology:

1. Socket: It is one end of inter-process communication channel (client server)

2. Types of Socket: There are different types of sockets. I used stream socket. Stream sockets use TCP (Transmission Control Protocol), which makes communication continuous and is best for a chatting app.

3. Address Domain: There are basically two domains: UNIX domain and Internet domain. I used Internet Domain in which two machines on the Internet communicate.

The steps involved in developing server.c source code are as follows:

1. First of all, we need to import suitable libraries which make socket programming easy. Now, we need to create socket, socket() system call is used to create socket. For communication to work, sockets of server-client have to be of the same socket type and in the same domain. Socket() system call returns integer value and is called socket file descriptor.

2. After that, we need to bind the socket to an address and that can be done from bind() system call. bind() takes three arguments: socket file descriptor, address to which it is bound and the size of the address to which it is bound.

3. Now, we need to listen for connections and that can be done by using listen() system call. The listen system call allows the app to listen for connections. It takes two arguments: socket file descriptor and size of the backlog queue.

4. Accept connection from client using the accept() system call. This call blocks until a client connects with the server. It

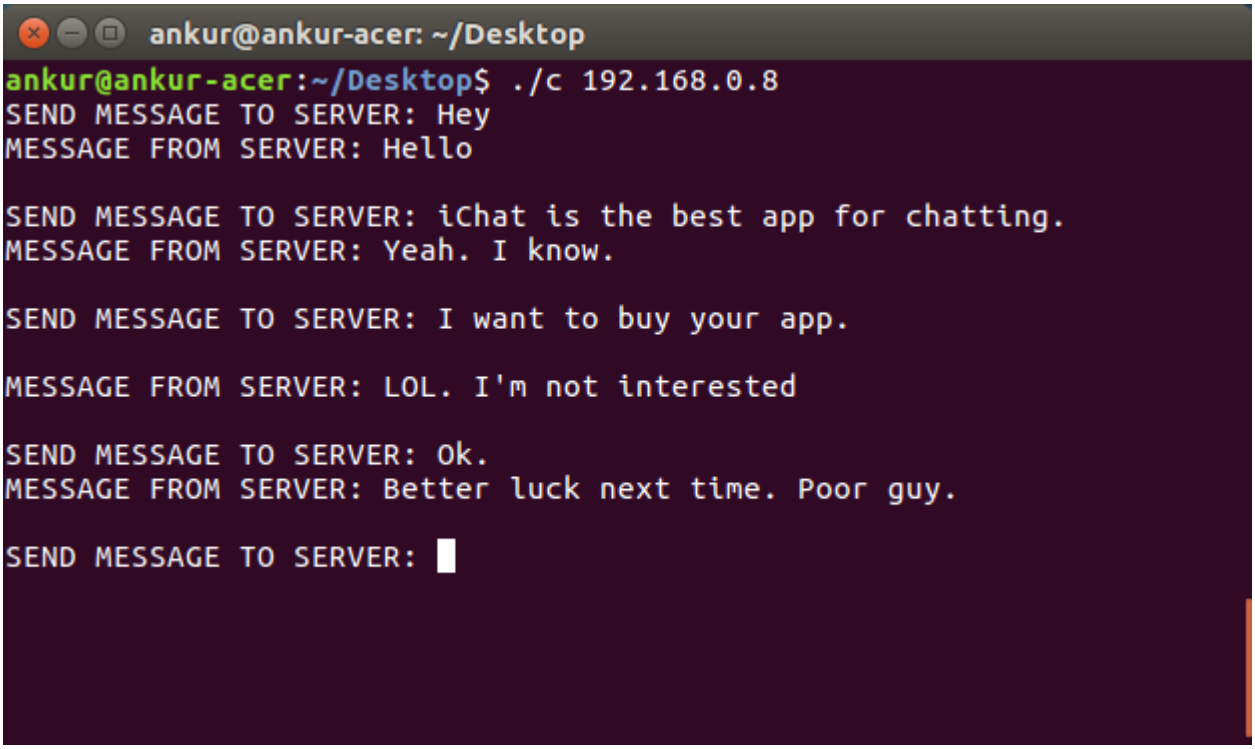
returns a new file descriptor, and all communication on this connection should be done using the new file descriptor.

5. Sending and receiving data: read() system call is used to read message from client. And write() system call is used to write message on client's terminal.

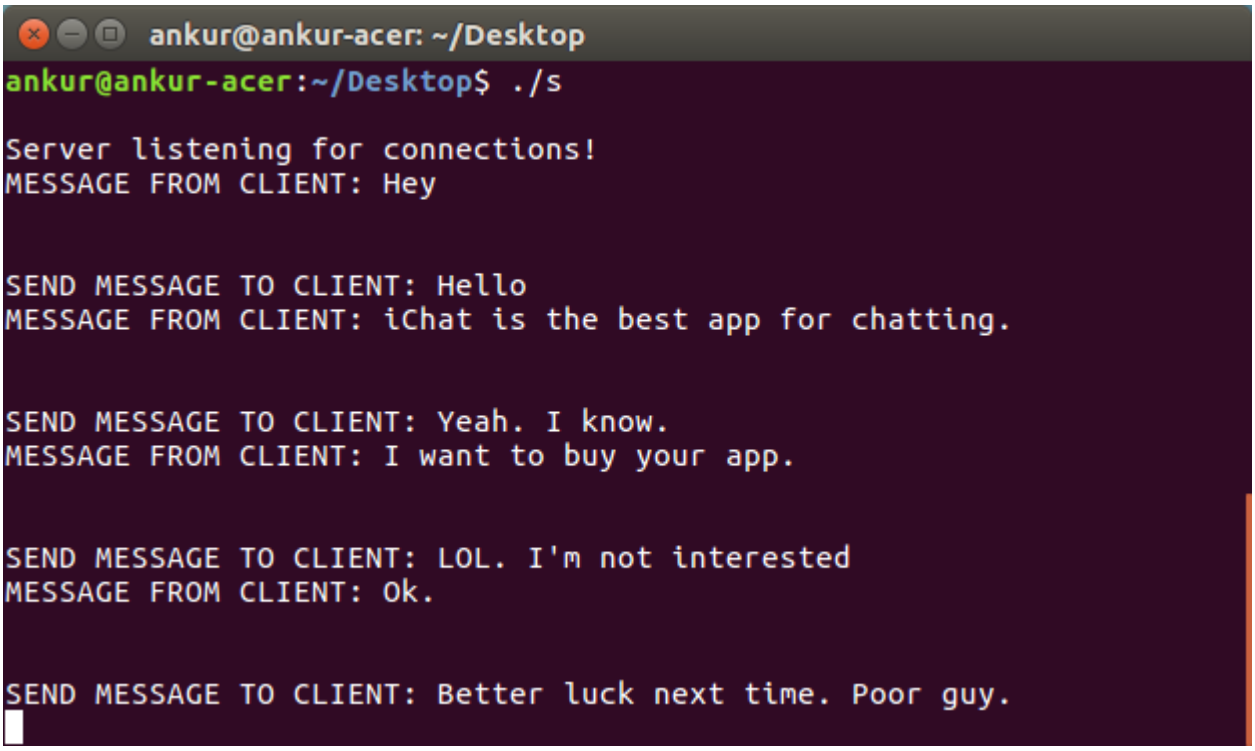
While executing client.c, we need to pass server's IP address to connect with server. The steps involved in developing client.c source code: Similar steps like server.c are required with some modification for client.c but we need to add some conditional statements when connection shows error.

Snapshots of "iChat":

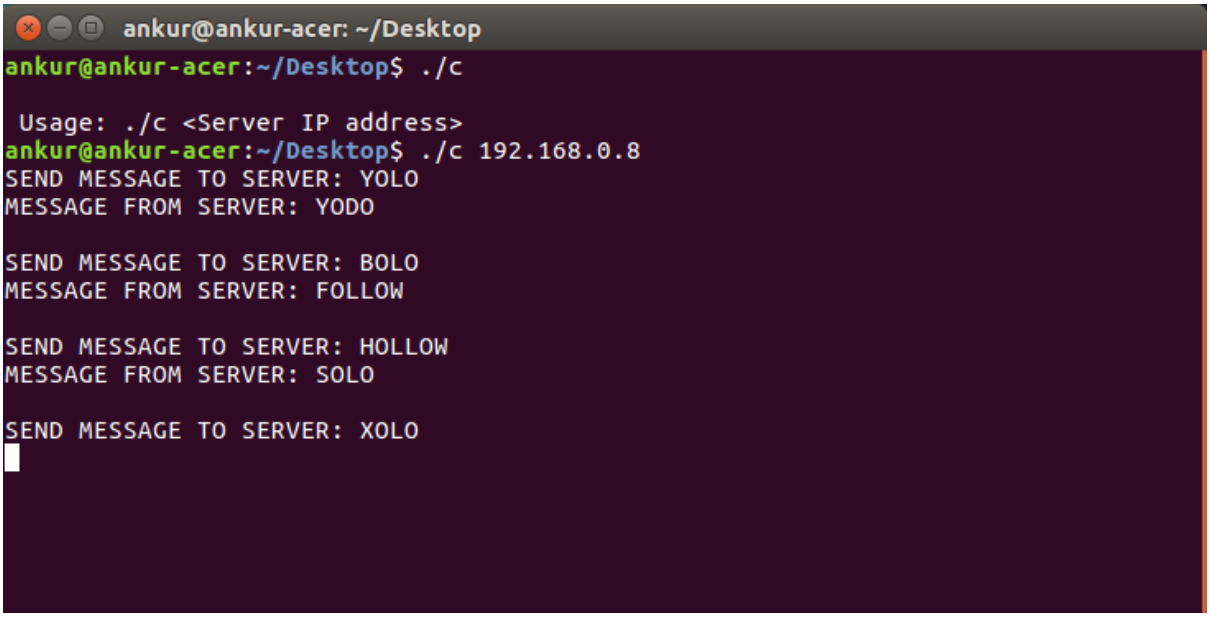
- Client Side Terminal-



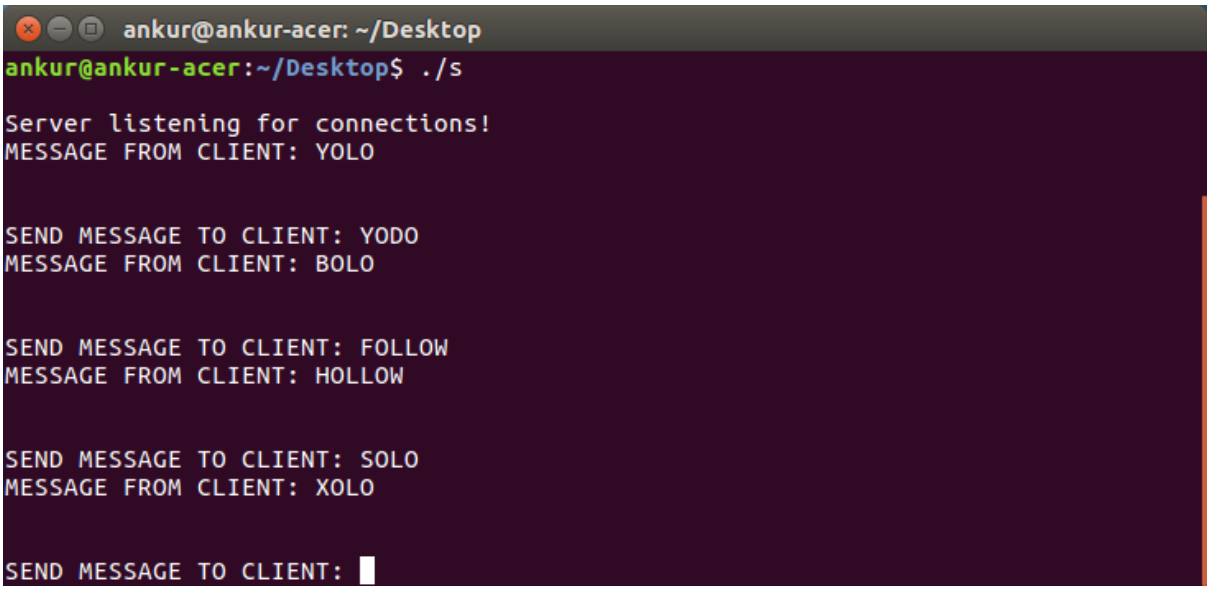
Server Side Terminal-



Client Side Terminal-



Server Side Terminal-



Contribution:

Coder, Researcher- Ankur Sharma (2016225)